

[Open in app](#)

## Parveen Khurana

124 Followers

[About](#)[Following](#)

# Expert Systems

 **Parveen Khurana** Nov 27, 2019 · 9 min read

In this article, we discuss Expert Systems(Rule-Based Systems) and how do we use Expert Systems for Binary Classification.

### Why do we care about Binary Classification?

The reason we care about Binary Classification is that in our Capstone Project, our main goal is that given an image containing some text, how do we identify this text and then translate/transliterate it to a language of our choice; that's the goal that we have. Now before we go to this final goal, one intermediate goal that we have is that, given an image, we should be able to find out if it contains text or not. So, that's a typical Yes or No problem which we call a Binary Classification problem. So, this is why we are interested in Binary Classification and to begin with this article, we will look at something known as an Expert System or Rule-Based system for Binary Classification.

[Open in app](#)

Example input image

Our goal eventually is to train a machine to make decisions for us; so a machine is going to look at an image and tell us whether it contains text or not but before we discuss that let's look at How do Humans make Decisions.

## How do humans make decisions?

So, to discuss this, let's take an example from the medical domain where a Doctor is looking at some patients and is trying to decide whether that patient has Dengue or not. So, the Doctor might be looking at several factors like Skin Rash, Fever, headache, cold cough, vomiting and so on and then decide whether that patient has dengue or not.

*How do humans make decisions?*

# Human decision making



skin rash



fever



headache














cold cough





Dengue

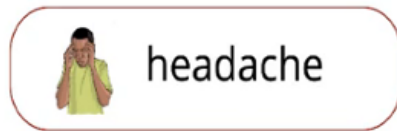
Let’s say below is the situation for one of the patient:

 skin rash		 Dengue
 fever		
 headache		
 cold cough		
 vomiting		

That patient had a Skin rash, did not have a fever, had a headache, did not have a cold cough and had vomiting and so on and based on this the doctor decides that this person does not have dengue.

Now in the case of another patient, a different combination of features was exhibited, some symptoms were there(present) some symptoms were not there but since most of the symptoms were there, it was enough for the Doctor to conclude that this person has dengue.

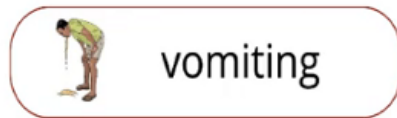
 skin rash	
---	---

[Open in app](#)


headache



cold cough



vomiting



Dengue

The Doctor is making this decision based on past experience, so in years of practice we can say that the Doctor has examined several patients/cases (for ex. data is shown for 4 patients below), for each of the cases, doctor had access to some specific set of symptoms (like skin rash, fever etc.) and also the final outcome i.e whether that patient was diagnosed with dengue or not.

Inputs					Output	
Rash	fever	headache	cold cough	vomiting		

So, based on access to a lot of such past data from personal experiences, the doctor is able to look at these symptoms and make a decision whether the patient has or does not have dengue.

Open in app

numbers)

inputs






Rash

fever

headache


cold cough

vomiting





1	0	0	0	0
1	0	0	1	0
1	1	1	0	1
1	0	1	1	0

output



0
0
1
0

	0
	1

Now the data is converted to a machine-readable input.

This type of decision making appears in all the domains and is not limited only to the medical domain. For example, in sports wherein cricket there is an umpire who needs to decide whether a person is out or not and let’s look at a specific case where the umpire needs to decide whether the player is LBW or not. Now, this again depends on various factors like where the ball was pitching, whether the impact was in line or not, what was the height of the delivery at that time and so on, and all these factors act as the input for the umpire to make the decision.

Inputs

Pitching in line

Impact

Height

no ball

shot attempted

LBW out



Output



Open in app



And once again we could make this input as machine-readable as below:

Inputs

Pitching in line

Impact

Height

no ball

shot attempted

Output

LBW out

1	1	0	1	0	0
1	1	1	0	0	1
1	1	1	0	1	1
1	0	1	1	0	0

✗

0

✓

1

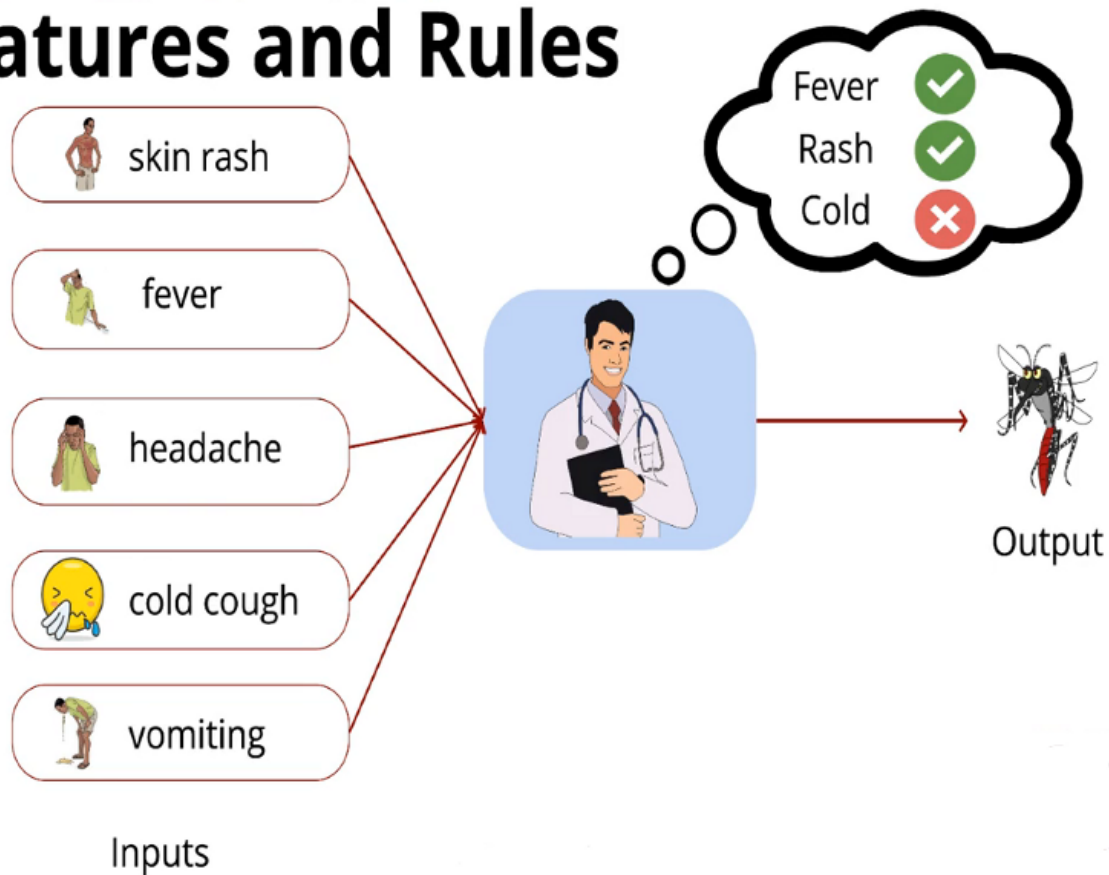
So now the question is that what is it that we are using to make decisions? So, now we want to understand the semantics of decision making and there are two things that play a role there, one is **features** and the other one **rules**.

**Features** are nothing but all these **inputs** that we are taking(symptoms in case of dengue example) and based on these inputs there is something going on in the mind of the doctor where he tries to combine these various inputs into some kind of rule-based

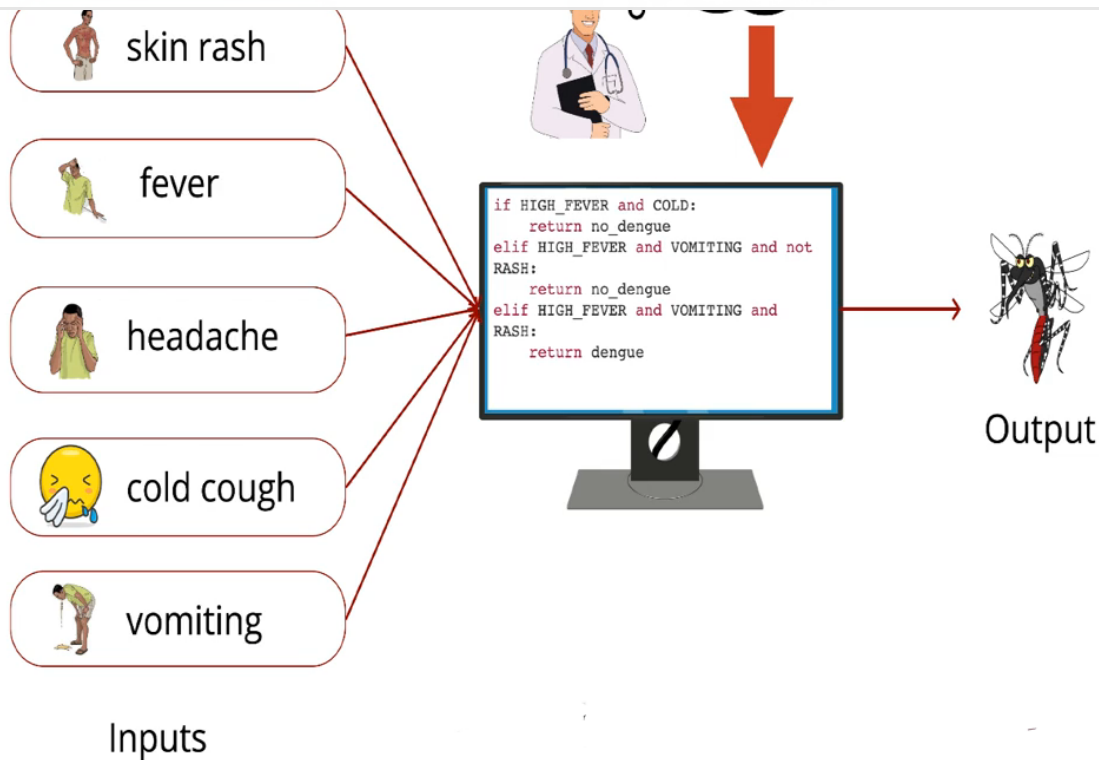
[Open in app](#)

dengue.

# Features and Rules



We want to outsource this task to a machine, so instead of a human making this decision by various rules in his/her head, we want to be able to translate this task to a machine in the form of a program and the machine should be able to execute this program by taking certain inputs and then give us an answer. So, to write out the rules for the Expert System, we take into account the various rules and code them out using the if-else code blocks.

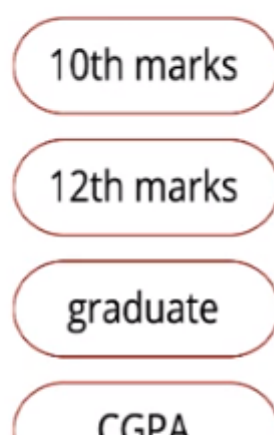
[Open in app](#)

So, this is what an Expert system looks like, its a conversion of all rules which are there is a human's head into a nice program which can then be executed and return an output.

## Limitations of Expert System

Let's take the task of hiring someone for a job. Now for that, we will look at various parameters like marks in 10th standard, 12th standard, whether Graduate or not and if yes what was the CGPA and so on.

### Task of hiring someone for a job

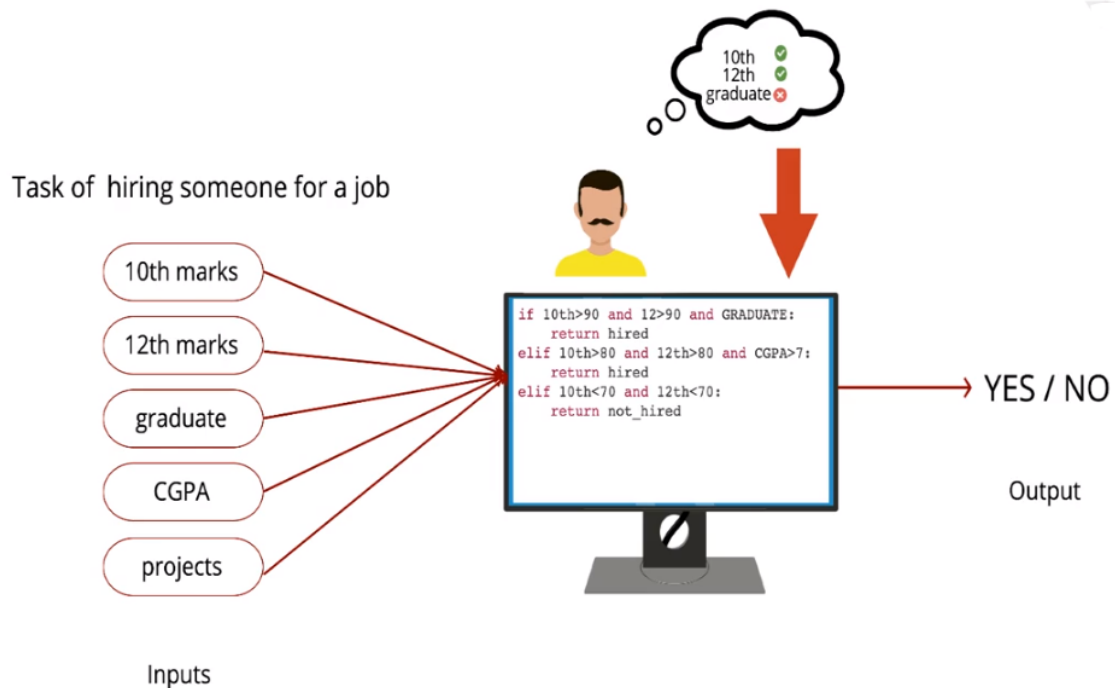




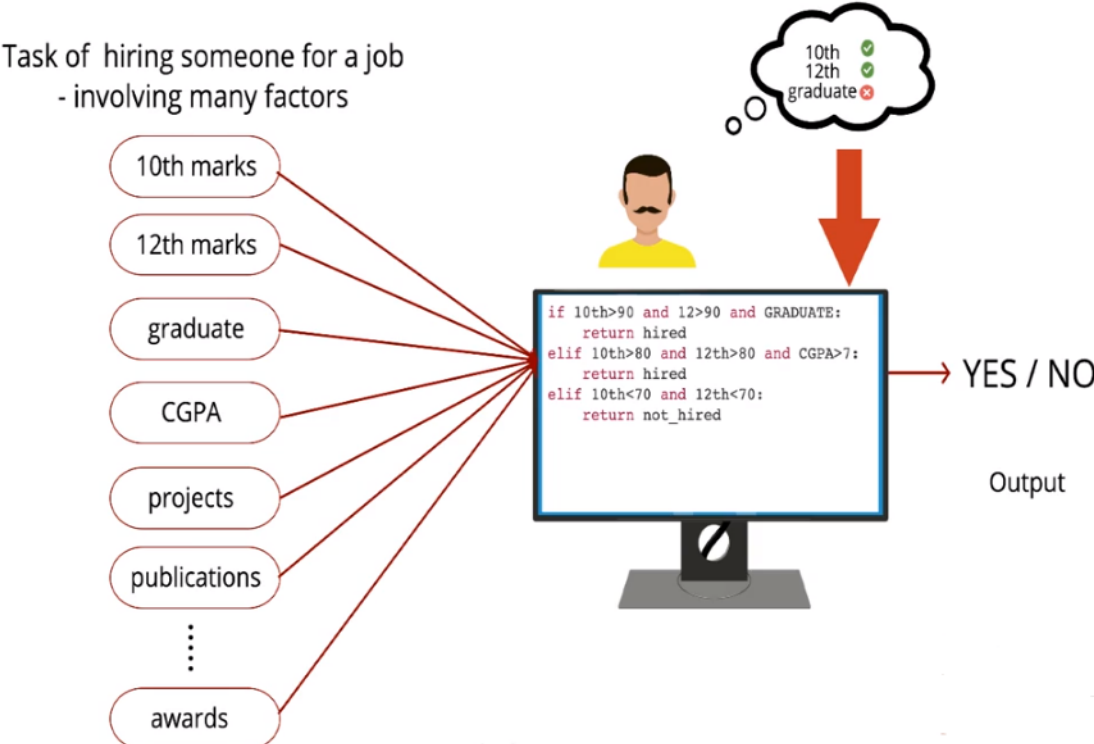
[Open in app](#)


## Inputs

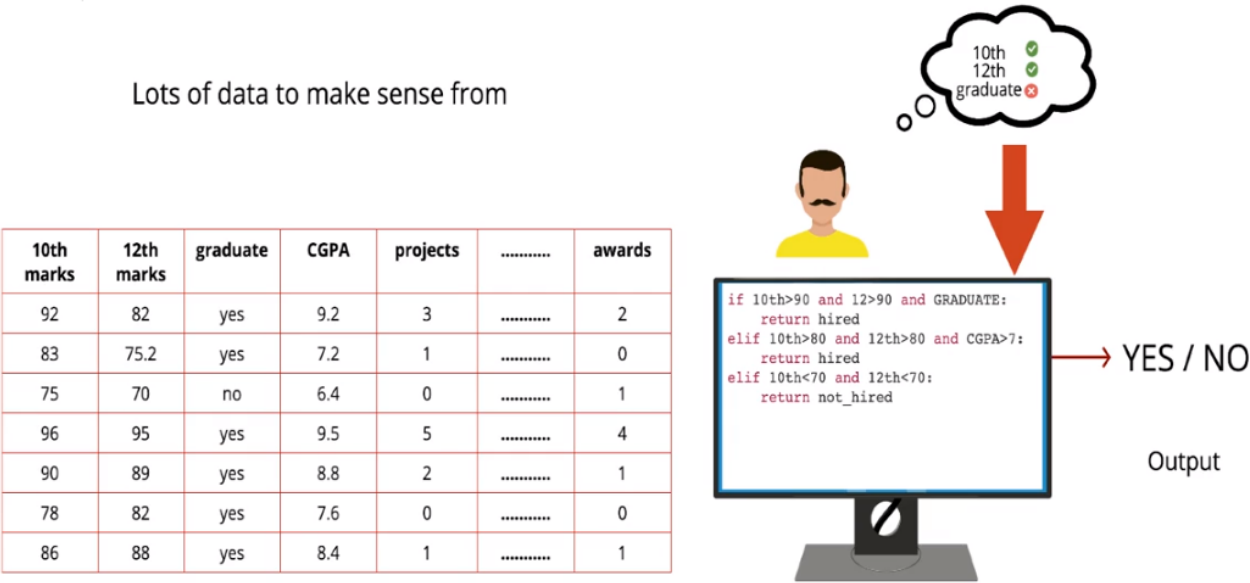
And based on these inputs, the hiring manager would have certain rules in their head that if a certain set of conditions are satisfied, then the person is hired. If these sets of conditions are not satisfied then the person is not hired. So, we can think of an Expert System in this limited constrained setting.



Now what happens in various practical scenarios including this very scenario (where we are considering this task of hiring someone for a job), we are loaded with a lot of data or we might have something known as many many factors, we do not base our decision on 4–5 inputs (as in above image), but there might be many other factors like does the person has any publications in the past, languages which the candidate is comfortable with, what are the database systems (for DB job) the candidate knows, are there any awards the candidate had received in the past and if yes then were these awards from



Now if we just list out all these data, it would look like as in the below image



Here in the above image, every row represents data for one employee that this

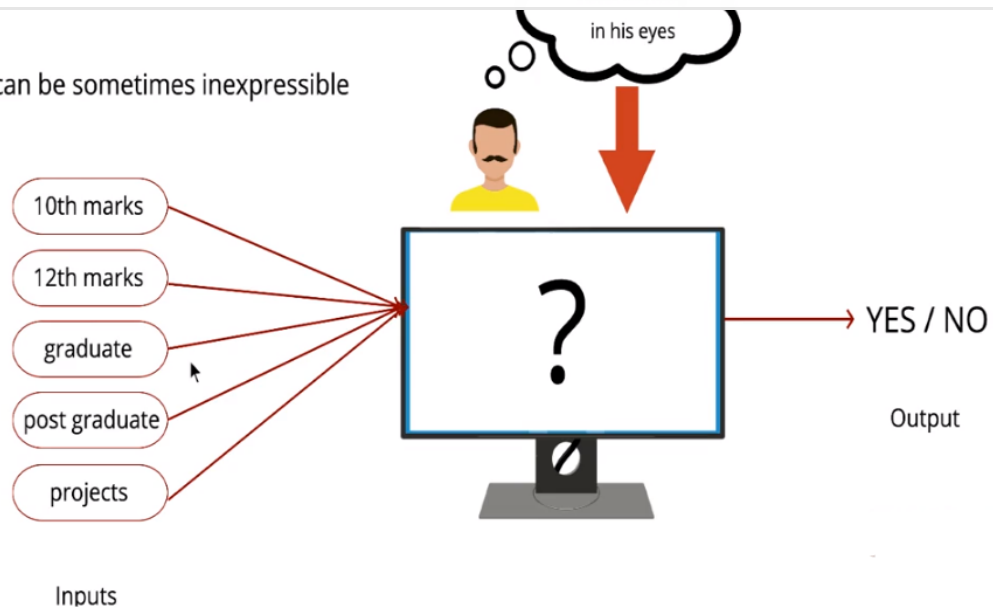
[Open in app](#)

whether this person did a good job(had a good career) in the company or not. Now what we want to do is that based on the available data, come up with some rules such that if combine these parameters in certain if-else conditions(if required we could also have weights for these conditions for example if CGPA is more important than anything else or knowledge of Java is more important than anything else etc.) that tells whether the company should hire a new person who has certain values for these features, so that's the job we are trying to do.

But the problem is, for any company of a typical size, they would have thousands of employees hired and fired and iteration and so on. A lot of people would have actually left the job, so we would have data for many employees and for each employee, we would have a large number of features. So, this leads to **a lot of data** which becomes hard for a human to make sense(for example how would you relate an experience in C or Java and relate to CGPA, what's the connection to 10th standard marks, how would you relate these in certain if-else conditions). So, **in any of the real-world problems, the first challenge is that we have lots of data to make sense of and it becomes hard for a human to break it down into certain if-else conditions. The second is, even if we somehow visualize all the data, the rules you might need to come up with might be very complex to write that down.** It would be so many permutations and combinations and it would become impossible at some point to remember everything and write it down. **So, rules may be just too complex to be able to write them down. In some cases, the rules may just be inexpressible for example, a hiring manager in addition to looking at all the quantitative matrix, there might be something else going on in his/her mind like they might say that I hired the candidate because I saw honesty in this candidate. So, how do we express this as rule?** How do we even quantify something like honesty? It's hard to write this as a rule.

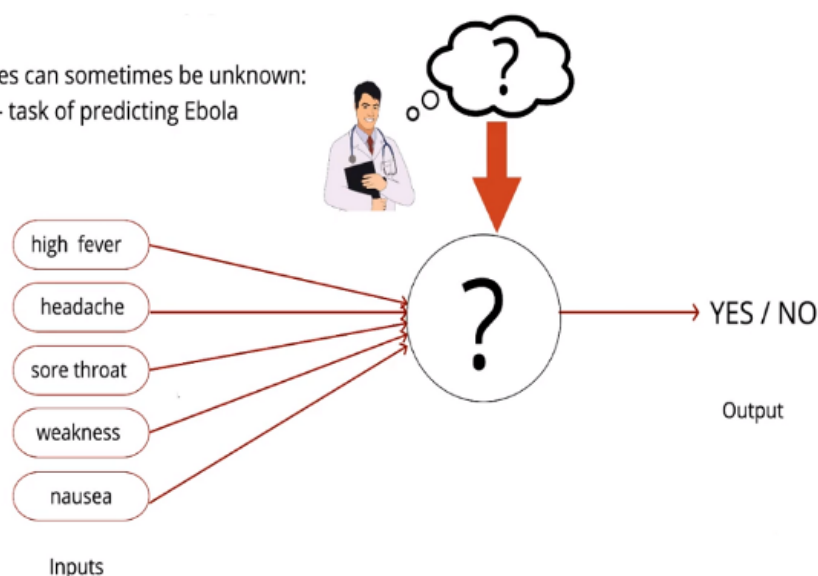
[Open in app](#)

The rules can be sometimes inexpressible



**The fourth challenge is that sometimes the rules may just be unknown**, example no matter how much data we have about Ebola, we still don't know the factors causing this disease, so we look at a lot of symptoms, but a lot of patients are exhibiting very different symptoms, there might be some common thread between these patients causing this disease or there might not be common thread between these patients, its very random and we don't know what leads to Ebola outbreak or why a person catches this disease or virus. So, the rules even a human does not know in that case despite having access to a lot of data.

The rules can sometimes be unknown:  
- task of predicting Ebola



[Open in app](#)

So, these are four challenges that typically any large scale rule based system actually faces:

- i.) Lots of data
- ii.) A lot of complex rules which are hard to write down
- iii.) And sometimes the rules themselves are not expressible meaning we can't even write them as a Boolean condition.
- iv.) Rules being unknown.

So, this led to the development of **Machine Learning** which is discussed in this [article](#).

This article covers the content covered in the Expert Systems — 6 Jars module of the [Deep Learning course](#) and all the images are taken from the same module.

[Machine Learning](#)[Expert Systems](#)[Ai Expert Systems](#)[Padhai](#)[One Fourth Labs](#)[About](#) [Write](#) [Help](#) [Legal](#)

Get the Medium app

